3.1. Función a inimizar en Python

```
def min_sq(x0,I_exp):
    """ Calculates squared differences for a minimization procedure.

    Given the experimental meassures of intensity and arbitrary values for
    the parameters of an SLM Jones Matrix, this function gives a value to
    minimize. That value tells how close is the estimation of x, y, z, w
    to the value that correctly models the SLM.

    :param x,y,z,w: Are a guess of real scalars that conform the Joung Matrix
     for the SLM.
    :param I_exp: Is a dictionary contanining intensities for every
    polarization state.
    """
    # brakets is a dictionary containing each pair of Jones vectors
    brakets = {1:translate_Ellipse_to_Jones([ 0,    0],       [0,0]),\
           2:translate_Ellipse_to_Jones([ pi/2,0],       [0,0]),\
           3:translate_Ellipse_to_Jones([ pi/4, 0],     [pi/4,0]),\
           4:translate_Ellipse_to_Jones([-pi/4, 0],     [pi/4,0]),\
           5:translate_Ellipse_to_Jones([pi/4,-pi/4],    [pi/4,-pi/4]),\
           6:translate_Ellipse_to_Jones([ -pi/4,pi/4],   [pi/4,-pi/4])}
    [x,y,z,w] = x0
    M = matrix([[ x + y*1j, z + w*1j],\
                [-z + w*1j, x - y*1j]])
    min_sum = 0
    I_sim = {}
    for i in range(1,nMeasures):
        Out, In = brakets[i]
        I_sim[i] = (In.H*M.H*Out * Out.H*M*In)
        min_sum += ((I_sim[i]-I_exp[i])**2)[0,0].real
    return min_sum
```